

Case-Based Policy and Goal Recognition

Hayley Borck¹, Justin Karneeb¹, Michael W. Floyd¹,
Ron Alford^{2,3}, and David W. Aha³

¹ Knexus Research Corp.; Springfield, VA USA

² ASEE Postdoctoral Fellow

³ Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5514); Washington DC; USA
{hayley.borck,justin.karneeb,michael.floyd}@kexusresearch.com
{ronald.alford.ctr,david.aha}@nrl.navy.mil

Abstract. We present the Policy and Goal Recognizer (PaGR), a case-based system for multiagent keyhole recognition. PaGR is a knowledge recognition component within a decision-making agent that controls simulated unmanned air vehicles in Beyond Visual Range combat. PaGR stores in a case the goal, observations, and policy of a hostile aircraft, and uses cases to recognize the policies and goals of newly-observed hostile aircraft. In our empirical study of PaGR’s performance, we report evidence that knowledge of an adversary’s goal improves policy recognition. We also show that PaGR can recognize when its assumptions about the hostile agent’s goal are incorrect, and can often correct these assumptions. We show that this ability improves PaGR’s policy recognition performance in comparison to a baseline algorithm.

Keywords: Policy recognition, intelligent agents, goal reasoning, air combat

1 Introduction

The World War I and World War II dogfighting style of air combat is mostly obsolete. The modern capabilities of aircraft instead facilitate a longer range style of combat deemed *Beyond Visual Range* (BVR) combat. In this paper, we focus on how an Unmanned Aerial Vehicle (UAV) may participate in BVR combat operations. In BVR combat, significant time between maneuvers allows a UAV more time to plan its reactions. Accurately recognizing the actions and plans of other agents greatly reduces the complexity for planning in adversarial multi-agent systems [1]. However, long-range aerial observations are generally restricted to the basic telemetry data returned by radar (i.e., position and speed). This makes accurately recognizing an opponent’s actions difficult [2].

In previous work, we described a reasoning system called the *Tactical Battle Manager* (TBM), which uses goal reasoning [3, 4], policy recognition [5], and automated planning techniques to control a UAV in a simulated BVR environment. The existing policy recognition system, like most, uses observations to

directly predict the tactics of the opposing agents. However, the choice of tactic (or plan) for an agent is driven by its over-arching mission (or goal), which cannot be directly observed. For example in a defensive mission, maintenance of stealth and prevention of losses may drive the enacted policies of the agent. In contrast, aggressiveness and firing missiles at opponents may be more important in an offensive mission.

In this paper, we introduce the Policy and Goal Recognizer (PaGR; Section 4), which leverages agent goals in an attempt to improve policy recognition accuracy. PaGR is given a mission briefing, which includes the details of the upcoming mission such as the expected capabilities of the hostiles and their expected goal. PaGR’s cases contain the goal, observations, and policy of a hostile agent. During the mission, the goal and current observations are used in the similarity calculation to retrieve a newly-observed agent’s predicted policy. Our experiments in a simulated BVR environment (Section 5), involving 2v2 scenarios (i.e., 2 allied agents vs. two hostile agents), show that case retrieval is significantly more accurate when goal information is employed.

However, mission briefings may not be accurate, and our experiments also show that making incorrect assumptions about an adversary’s goal may negatively impact retrieval performance. To counteract inaccurate goal assumptions, PaGR leverages past observations to recognize when there is a discrepancy between the goals provided by the mission briefing and the actual goals of the hostiles. Once a discrepancy is recognized, PaGR then attempts to predict a goal that more correctly describes, and thus predicts, hostile agent intentions. Our experiments show that goal revision can reduce most of the negative impacts from incorrect briefings.

We begin with a discussion of related work (Section 2), followed by a description of the TBM (Section 3). We then describe PaGR (Section 4) and our empirical study of it (Section 5), and finish with a discussion of future work (Section 6) and conclusions (Section 7).

2 Related Work

Plan, activity, and intent recognition has been an active field of research in AI in recent years [6]. Our system endeavors to recognize the tactics of opposing agents. We define an agent’s tactics as a sequence of actions it performs as part of a mission scenario, with the possibility it will switch tactics during a scenario. In this sense, an adversarial agent uses a plan to perform a tactic while having the autonomy to change tactics as necessary (e.g., because a previous tactic was unsuccessful, as countermeasures against another agent, it changed its goals).

Vattam et al. [7] use action sequence graphs to perform case-based plan recognition in situations where actions are missing or noisy. Ontanón et al. [8] use case-based reasoning (CBR) to model human driving vehicle control behaviors and skill level to reduce teen crash risk. Their CBR system predicts the next action the driver would take given the current environment state. However, these

systems cannot identify plans that were previously identified incorrectly and refine the plan recognition process.

Fagundes et al. [9] focus on determining when it is appropriate or necessary to interact with agents to gain more information about their plan. Their system implements symbolic plan recognition via feature decision trees to determine if an agent should interact with other agents in multiagent scenarios. Unlike the BVR domain, they assume full observability of other agents. Similar to Fagundes et al., we have shown previously that acting intelligently via automated planning significantly reduces the time it takes to classify agent behaviors in single-agent scenarios [10]. Although we do not perform active recognition in this paper, we extend its recognition system by adding an agent goal to the case.

Laviers and Sukthankar [11] present an agent that learns team plan repair policies by recognizing the plans of the opposing American football team. Molineaux et al. [12] also study plan recognition in this domain; their system uses recognized plans to aid in case-based reinforcement learning. Similarly, human activity recognition [13] has been used to recognize a senior citizen's activity and adapt an agent to be more helpful when assisting the senior. These efforts are similar to our own in that they attempt to revise the model of other agents. However, unlike our agent in the BVR domain, these recognition agents have access to perfect information.

Single-agent keyhole plan recognition can be expanded to multiagent domains, but has been shown to be a much more difficult problem [14]. Zhou and Li [15] show that multi-agent plan recognition can be performed in a partially observable environment by representing team plans as a weighted maximum satisfiability problem. However, their algorithm's runtime is proportional to the amount of missing information which could prove problematic in a real-time combat domain like BVR. As such, we instead determine the probability of each tactic being used. Our system can then assume that a specific tactic is being used or perform exploratory actions that will help differentiate between possible tactics.

Plan recognition can also assist in behavior or intent recognition. Geib and Goldman [16] describe the need for plan and intent recognition in intrusion detection systems. Their system operates in a partially observable environment with a single adversarial agent. They use a probabilistic plan recognizer that reasons about observed actions, generates hypotheses about the adversary's goals, and retrieve plans from a plan library. Their system operates under the assumption that an attacker targets the agent's system. In our work, we instead use plan recognition to identify an adversary's plans and the target of those plans (e.g., is the hostile attacking the agent or one of its teammates).

Corchado et al. [17] present an agent that encodes the belief-desire-intention (BDI) model in a CBR context. Their agent uses its current beliefs and desires (the problem) to select its intentions (the solution). This is similar to our own work in that it explicitly encodes the desires of an agent in cases. However, our cases are created from observations of adversarial agents, so they may contain erroneous or incomplete information about beliefs, desires, or intentions.

Uncertainty about the quality (e.g., due to sensing errors) or completeness (e.g., due to partial observability and hidden internal information) of observed information has been a focus of CBR systems that learn by observation [18–20]. These systems learn to perform a task by observing an expert, either with or without the expert’s knowledge and assistance. In our system, we are not trying to replicate an agent’s behavior but instead use observations to identify the agent’s behavior and use this to guide our agent’s decision making process.

Traces of a user’s behavior have been used as cases in trace-based reasoning [21] and episodic reasoning [22] systems. These systems store a sequence of the user’s past actions, or both past actions and past environment states. This allows the systems to reason over the current problem, the problem’s evolution over time, and the solutions that have been applied to the problem. These systems differ from our own in that we do not have direct knowledge of the observed agent’s action but must instead infer them from observable data.

3 Tactical Battle Manager

PaGR will serve as a component in the Tactical Battle Manager (TBM; Figure 1), a system we are developing for collaborative pilot-UAV interaction and autonomous UAV control. During experimentation and testing the pilot is modeled by using a static policy (see section 5). In its current state, the TBM contains, among other components, a goal reasoning component (*Goal Management System*), a planning and prediction component (*Planning System*), and a policy recognition component (*Policy Recognizer* in *Knowledge Updater*).

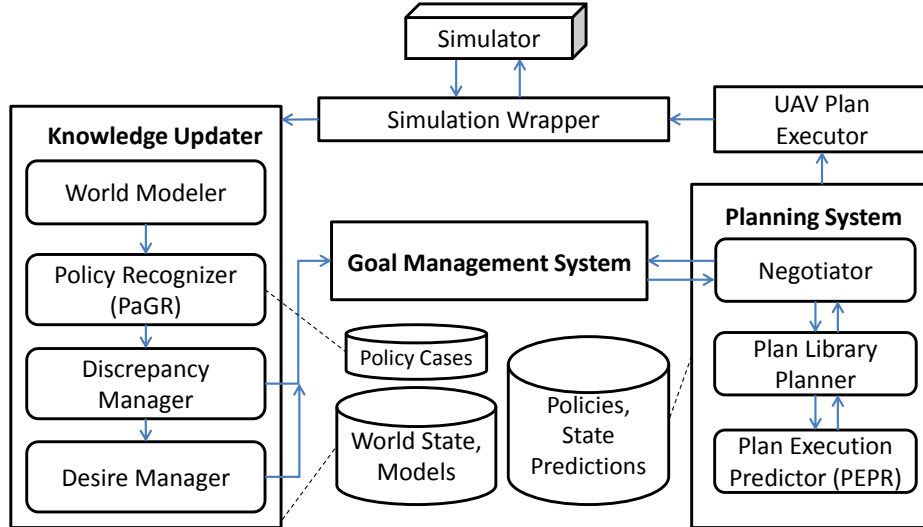


Fig. 1. The Tactical Battle Manager (TBM) architecture.

The goal reasoning (GR) component chooses between competing goals. In the TBM a goal is a list of weighted desires. Each desire has an agitation function that is updated based on the world state at each update cycle. The more a desire is agitated, the more urgently the agent wants to alleviate the symptoms (i.e., perform actions that reduce agitation). The GR component is not the focus of this paper. In our evaluations, a static goal is used by the UAV. In the future, the UAV will use observations, recognized hostile policies, and goals provided by PaGR to revise the current goal.

Our planner is an extension of a simple plan library planner [23]. The Planning System chooses a policy from a fixed set of eight candidates, which are either one of four basic movements (listed immediately below) or one of these four movements where the agent also fires a missile.

- **Pure Pursuit:** An agent flies directly at a target.
- **Drag:** An agent flies directly away from a target.
- **Beam:** An agent flies at a 90° offset to a target.
- **Crank:** An agent flies at the maximum offset but tries to keep its target observable in radar.

The missile-firing variants of these policies are: *ShootPursuit*, *ShootDrag*, *ShootBeam*, and *ShootCrank*. While these policies are simple, they encompass a range of real-world BVR combat tactics as described by our subject matter experts.

The planner uses the Plan Execution Predictor (PEPR) [24] to generate potential future states for all eight policies (i.e., what would happen if the UAV used each of those policies). These policies are then ranked by the weighted desire agitation over the duration of the prediction (i.e., how well the UAV’s goals are met), and the highest-ranked policy is returned. For example, if the current goal has a highly weighted safety desire, the planner will favor policies that do not put the agent in danger (i.e., Drag). If the safety desire is only somewhat agitated, the planner may choose a safe policy that also satisfies its other desires, such as ShootBeam.

Since the outcome of our policies (and thus desire agitation) depends on the actions of the other agents, the planner needs to have accurate information about the policies of opposing agents. PaGR provides this information.

4 Policy and Goal Recognition for BVR Combat

PaGR uses a three-part case structure to effectively recognize the goal and policy of other agents in a scenario. In our previous work on the Case-Based Behavior Recognizer (CBBR) system [2], cases were partitioned into two parts: observations of the agent and the agent’s policy. PaGR adds a third part, the agent’s goal. Case similarity is computed using either the observations and the goal, or just the observations (i.e., when trying to revise the goal). A typical CBR cycle generally consists of four steps: retrieval, reuse, revision, and retention. Currently, PaGR implements only the retrieval and reuse steps.

Table 1. Example Case within PaGR

Component	Representation
Goal	A list of weighted desires - 0.25 Aggressive Posture - 0.50 Safety - 0.25 Fire Weapon
Observations	A set of features discretized from the observable world state - 1.00 FacingTarget - 0.45 TargetRange - 0.15 InDanger
Policy	A set of ungrounded actions - Fire Missile - Fly Drag

4.1 Case Representation

Each case in PaGR represents a hostile agent and its potential interactions with one agent of the friendly team in the scenario. During case authoring, we created a case for each pair of friendly and hostile aircraft in the scenario based on their interactions. We will later use these cases to determine the most likely target of a hostile agent.

A case $C = \langle G, O, \Pi \rangle$ has three components (see Table 1). First, the *goal* G is a list of the weighted desires of the agent. Second, the *observations* O of the position and heading of the hostile agent are recorded. These observations are then projected onto features such as FACINGTARGET (the relative bearing to the target), TARGETRANGE (a normalized distance between the agent and its target), and INDANGER (whether the agent is in the weapon range of the target). Each feature, normalized to between 0 and 1, represents how much the feature is currently exhibited by the observations. For example, FACINGTARGET will be 1 if the hostile agent is directly facing the target but closer to 0.5 if it is facing the target at a 90° angle. Finally, a *policy* Π is recorded as a sequence of ungrounded actions that define an air combat tactic. We use policies rather than plans, which have grounded actions, to help PaGR overcome the missing information that is prevalent in this domain. As stated previously, which parts of the case represent the problem and which represent the solution change depending on PaGR’s current task (i.e., policy identification or updating goal assumptions). The possible observational features are listed below:

1. Facing Target
2. Closing on Target
3. Target Range
4. Within a Target’s Weapon Range
5. Has Target within Weapon Range
6. Is in Danger
7. Is moving with respect to Target

4.2 Similarity and Retrieval

The similarity and retrieval steps of PaGR are relatively straightforward. During the recognition of a policy, similarity is calculated between the goal and observations of the current query q and those of each case c in the case base CB .

The similarity of two observations is defined as the average distance of the matching features (Equation 1), where $\sigma(w_f, q_f, c_f)$ is the weighted distance between two values for feature f and N is the set of time step features.

$$sim_o(q, c) = \frac{\sum_{f \in N} \sigma(w_f, q_f, c_f)}{|N|} \quad (1)$$

The similarity of two goals is defined as the distance of the weights of the desires that define the goal. If a desire is missing (e.g., q has desire $f \in D$ but c does not) then it is treated as having a weight of 0 (Equation 2).

$$sim_g(q, c) = \frac{\sum_{f \in D} (w_f, q_f, c_f)}{|D|} \quad (2)$$

The similarity between two cases is then defined as the weighted average of the observation and goal similarities (Equation 3).

$$sim(q, c) = w_o \times sim_o(q, c) + w_g \times sim_g(q, c), \quad (3)$$

where $w_o + w_g = 1$. Using this equation for case similarity, PaGR retrieves a set of cases \mathcal{C}_q , where each case in the set has a similarity greater than a given threshold parameter τ_r . If no cases were retrieved over the threshold, the policy is marked as unknown and system continues until the next timestep to repeat the retrieval process with more information. After it retrieves \mathcal{C}_q , PaGR returns a normalized ratio of the policies in the cases of \mathcal{C}_q that represents their relative frequencies with respect to their case weights generated during pruning. We interpret this as a probability distribution, so that given a policy p with a ratio of 0.7, there is a 70% chance that this is the policy being executed.

4.3 Pruning Strategy

We perform a limited amount of case base maintenance (CBM) on our case library. In particular, we prune cases in a manner that is similar to the strategy we used in our previous work [10], although now designed for our revised three-part case structure. As we have done previously, our pruning algorithm merges similar cases that have the same policy while maintaining their overall importance. That is, whenever two cases with the same policy have a similarity over a certain threshold τ_π , we merge them and increment an associated counter. After all similar cases have been merged, an additional sweep is made over the (now pruned) case base. Each case has its weight normalized with respect to all cases sharing its policy. This prevents over-abundant cases from overpowering less common ones during retrieval. The resulting case base is smaller than the original but preserves the differences between prominent and atypical cases.

4.4 Using PaGR in BVR Mission Scenarios

Before each simulated 2v2 scenario is run, a mission briefing is given to the UAV for use by PaGR. This mission briefing includes details on the expected goal of the hostiles, and a case base (described in Section 4.1). For example, one of the mission briefings used in our experiments informs the UAV that all encountered enemy aircraft will have highly aggressive goals and do not care about their own safety.

During the simulation, PaGR can be configured to use the expected goal in one of three ways: (1) ignore it (a baseline approach), (2) use the given expected goal during case retrieval, or (3) perform goal revision and, during retrieval, use whatever goal it thinks is most likely being pursued.

PaGR performs goal revision by searching for the goal that best explains all prior observations. For each such observation, PaGR retrieves all the cases \mathcal{C}_q whose similarity is above τ_r (ignoring their goals). Just as with policy retrieval, PaGR creates a probability distribution over the retrieved goals. These distributions are averaged over the duration of the scenario. During goal revision, the goal with the highest average probability is used as the new mission-briefing goal and is used for retrieving future policies.

5 Empirical Study

In our empirical study we examine the following hypotheses:

- H1:** Using an agent’s goal during retrieval will increase PaGR’s policy recognition accuracy.
- H2:** PaGR can effectively determine a goal discrepancy through observation.
- H3:** If given an incorrect goal, PaGR can revise it to increase policy recognition accuracy.

5.1 Scenarios

We created three scenarios and five mission briefings to test PaGR. Creation of the scenarios and briefings were guided by subject matter experts. These briefings specify goals for the opposing agents aircraft with varying levels of aggressiveness, namely:

- **HIGHLYAGGRESSIVE:** Approach directly and fire.
- **AGGRESSIVE:** Approach indirectly and fire.
- **SAFETYAGGRESSIVE:** Fire at range and leave.
- **OBSERVE:** Approach indirectly but do not fire.
- **SAFETY:** Do not approach.

All our 2v2 scenarios involve two *blue* allied aircraft flying into an airspace with two *red* hostile aircraft approaching from various angles (Figure 2). The blue aircraft are governed by a static **HIGHLYAGGRESSIVE** policy; they fly directly

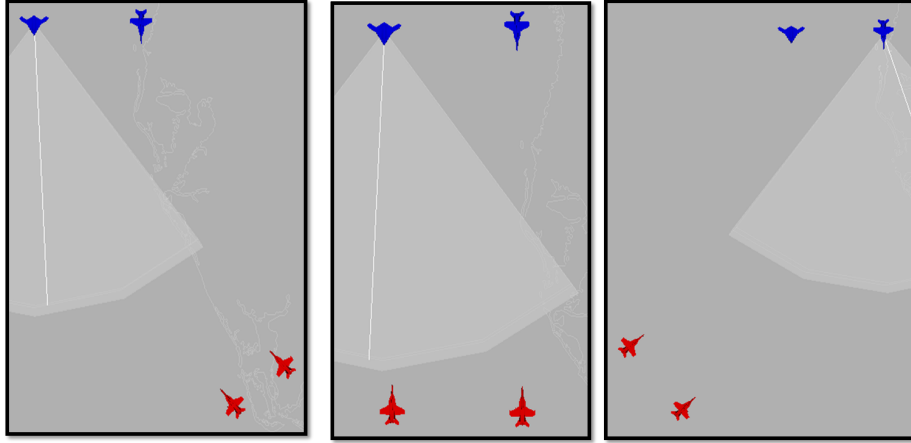


Fig. 2. Base 2v2 scenarios with friendlies in blue and hostiles in red. Lightened areas represent the radar observation cones of a single aircraft (other cones are not shown).

at their target and attempt to shoot a missile when in range. We run PaGR on only *one* of the blue aircraft (representing the UAV). We chose to not use a full TBM to control the second blue aircraft while testing PaGR so that the results would not be influenced by the TBM’s other components. The red aircraft are controlled by a simplified version of the TBM; they are given a static goal from the mission briefing to use with the plan library planner, and do not invoke PaGR. The red aircraft mark all opposing agents as using a basic policy that continues the current heading and speed for the purposes of prediction. Thus, in a given mission, the red aircraft will react to the blue aircraft using the goal weights specified in the mission briefing.

To create a large pool of test scenarios, we applied a random perturbation 33 times to each of the 15 ⟨scenario, mission briefing⟩ pairs, which yields a total of 495 variants. This scenario perturbation step modifies the headings and positions for each agent individually in the scenario within bounds to generate random but valid scenarios. By “valid”, we mean that one of the agents will eventually come within radar range of another agent. The mission briefings and scenarios were created to encompass a set of real-world scenarios as described by our subject matter experts.

5.2 Empirical Method

We used a stratified k -fold cross validation strategy, where $k = 10$, and split the variants evenly across each fold. For each (test) fold, a case base was created using the other (training) folds and pruned as described in Section 4.3. Finally, all variants contained within the current testing fold were run for evaluation purposes. Section 5.3 describes the experiments and algorithms used to test our hypotheses.

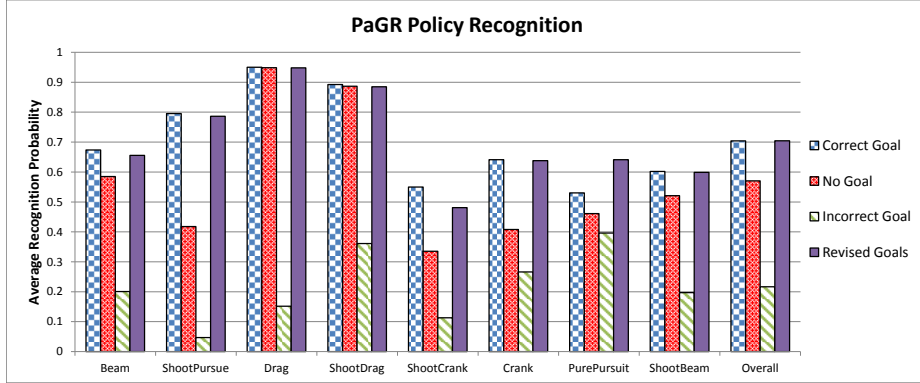


Fig. 3. Average policy recognition accuracy results from our first experiment showing the policy recognition accuracy of PaGR tested in three conditions

To test hypothesis **H1**, that using goals improves PaGR’s policy recognition accuracy, we tested PaGR in the following conditions.

1. **Correct Goal:** This was the ideal case for PaGR; it was given a correct goal from the mission briefing and never tried to change it. Additionally, when performing case retrieval it weighted goal similarity at 20% ($w_g = 0.2$) and observations at 80% ($w_o = 0.8$). Preliminary tests showed that the specific value for goal weighting had little impact on overall recognition as long as $w_g \in [15\%, 25\%]$.
2. **No Goal:** In this condition, PaGR did not use goals during recognition. That is, during case retrieval, it weighted goal similarity at 0%, meaning it used only observations to recognize policies.
3. **Incorrect Goal:** This tests PaGR in its worst-case scenario. This configuration was identical to the Correct Goal condition except it was always given an *incorrect* goal.
4. **Revised Goals:** We used this condition to test **H2** and **H3**. PaGR was given an incorrect goal, but in addition to the normal policy retrieval recognition, PaGR ran an additional retrieval using only observations. However, rather than returning a policy, it returns a probability distribution of goals. PaGR averages this probability across the entire scenario executed so far and checks whether the current goal is the most probable. If not, it reports a discrepancy that triggers goal recognition of a recognized goal.

To calculate the results of these runs, we computed the percentage of recognition attempts that PaGR returned the actual policy being enacted by the target agent. We averaged these values for each policy across all variants.

5.3 Results

Figure 3 displays the average results for each of the 8 policies and their average (*Overall*). The results support **H1**; in all cases where PaGR was given a cor-

rect goal, it performed at least as well as when it was run without using goals and in many instances it performed better. In the two cases where the systems performed similarly (Drag and ShootDrag), the policies are easily recognizable given the observations as they are the only plans that involve completely facing away from the target. We conducted an independent samples *t*-test to compare the results for the Correct Goal and No Goal conditions for all the policies. This yields *p* values in the range of 10^{-5} to 10^{-9} for all policies except Drag and ShootDrag. When run in the Incorrect Goal condition, PaGR performs far worse than the No Goal condition.

The results for our second experiment are displayed in Figure 4, where the blue bars denote PaGR’s ability to recognize that a given goal is incorrect and the red bars denote that it also identified the hostile agent’s true goal (i.e., the most probable goal was the true goal). PaGR’s discrepancy recognition is high; it almost always correctly identifies when the current goal is incorrect given the observations. This lends some support to **H2**. Figure 4 also shows that some true goals are easier for PaGR to recognize than others.

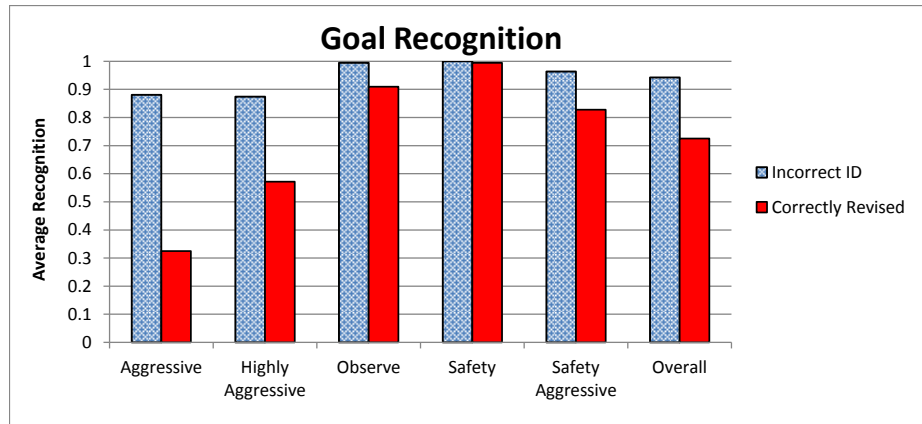


Fig. 4. Goal recognition accuracy results. The mission briefing goal is recognized as being incorrect (blue bars) if goal revision selects any other goal to replace it. The goal is correctly revised (red bars) if PaGR selects the true goal.

Finally, to test hypothesis **H3** we re-ran the Revised Goals condition used to test **H2**, except this time we allowed PaGR to replace the current goal with the most probable goal based on its averaged observations. Figure 3 displays the Revised Goals results. PaGR’s recognition probability when using goal revision is almost equivalent to when using the Correct Goal condition. At first glance these results may seem confusing because the Correct Goal recognition scores are not nearly high enough to always revise the goal correctly. Often times it seems that PaGR confuses two similar goals, such as **HIGHLYAGGRESSIVE** and **AGGRESSIVE**. However, because these two goals can often lead to similar policies,

PaGR’s overall correct recognition is high. A t -test comparing the Revised Goals and No Goal results yields p scores in the same range as we found for hypothesis **H1**. Therefore, this lends some support for hypothesis **H3**.

6 Future Work

In Beyond Visual Range air combat a hostile agent will often try and disguise their actions or true intent. Therefore, in future work we plan to extend PaGR to recognize deceptive agents in BVR scenarios. We also plan to integrate PaGR into the TBM. Leveraging PaGR’s ability to accurately predict hostile policies and goals should increase the TBM’s ability to effectively reason about goals and plans. We also plan to investigate how active techniques for policy recognition could impact PaGR’s results in comparison to our prior work on active behavior recognition [10]. Finally, we plan to show that using PaGR allows for overall improvements to mission performance in longer and more complex scenarios, compared to using no recognition system (or an alternative recognition system).

7 Conclusions

In this paper we presented the Policy and Goal Recognizer (PaGR). We tested PaGR in simple 2v2 Beyond Visual Range air combat scenarios, where one of the two friendly vehicles uses PaGR. We found that using PaGR’s prediction of a hostile agent’s goal increases its policy recognition accuracy (as compared to when using only its observations). In our scenarios, PaGR can effectively determine whether the mission briefing provided the correct goal for a hostile agent. We also found that, when using the same case structure and information used for policy recognition, PaGR can revise incorrect goal information and use goals that better fit the observations, which increases policy recognition accuracy.

Acknowledgements

Thanks to OSD ASD (R&E) for supporting this research. Thanks also to our subject matter experts for their many contributions and to the reviewers for their helpful comments.

References

1. Carberry, S.: Techniques for plan recognition. *User Modeling and User-Adapted Interaction* **11**(1-2) (2001) 31–48
2. Borck, H., Karneeb, J., Alford, R., Aha, D.W.: Case-based behavior recognition in beyond visual range air combat. In: *Proceedings of the Twenty-Eighth International Florida Artificial Intelligence Research Society Conference*, AAAI Press (2015)

3. Muñoz-Avila, H., Jaidee, U., Aha, D.W., Carter, E.: Goal-driven autonomy with case-based reasoning. In: *Proceedings of the Eighteenth International Conference on Case-Based Reasoning*. Springer (2010) 228–241
4. Molineaux, M., Klenk, M., Aha, D.W.: Goal-driven autonomy in a Navy strategy simulation. In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI Press (2010)
5. Borck, H., Karneeb, J., Alford, R., Aha, D.W.: Case-based behavior recognition to facilitate planning in unmanned air vehicles. In Vattam, S.S., Aha, D.W., eds.: *Case-Based Agents: Papers from the ICCBR Workshop (Technical Report)*. University College Cork: Cork, Ireland (2014)
6. Sukthankar, I.G., Goldman, R., Geib, C., Pynadath, D., Bui, H.: An introduction to plan, activity, and intent recognition. In Sukthankar, I.G., Goldman, R., Geib, C., Pynadath, D., Bui, H., eds.: *Plan, Activity, and Intent Recognition*. Elsevier (2014)
7. Vattam, S.S., Aha, D.W., Floyd, M.: Case-based plan recognition using action sequence graphs. In: *Proceedings of the Twenty-Second International Conference on Case-Based Reasoning*. Springer (2014) 495–510
8. Ontanón, S., Lee, Y.C., Snodgrass, S., Bonfiglio, D., Winston, F.K., McDonald, C., Gonzalez, A.J.: Case-based prediction of teen driver behavior and skill. In: *Proceedings of the Twenty-Second International Conference on Case-Based Reasoning*. Springer (2014) 375–389
9. Fagundes, M.S., Meneguzzi, F., Bordini, R.H., Vieira, R.: Dealing with ambiguity in plan recognition under time constraints. In: *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, ACM Press (2014) 389–396
10. Alford, R., Borck, H., Karneeb, J., Aha, D.W.: Active behavior recognition in beyond visual range combat. In: *Proceedings of the Third Conference on Advances in Cognitive Systems*, Cognitive Systems Foundation (2015)
11. Laviers, K., Sukthankar, G.: A real-time opponent modeling system for Rush Football. In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, AAAI Press (2011) 2476–2481
12. Molineaux, M., Aha, D.W., Sukthankar, G.: Beating the defense: Using plan recognition to inform learning agents. In: *Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference*, AAAI Press (2009) 337–343
13. Levine, S.J., Williams, B.C.: Concurrent plan recognition and execution for human-robot teams. In: *Twenty-Fourth International Conference on Automated Planning and Scheduling*, ACM Press (2014)
14. Banerjee, B., Lyle, J., Kraemer, L.: The complexity of multi-agent plan recognition. *Autonomous Agents and Multi-Agent Systems* **29**(1) (2015) 40–72
15. Zhuo, H.H., Li, L.: Multi-agent plan recognition with partial team traces and plan libraries. In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, AAAI Press (2011) 484–489
16. Geib, C.W., Goldman, R.P.: Plan recognition in intrusion detection systems. In: *Proceedings of the DARPA Information Survivability Conference*, IEEE Press (2001) 46–55
17. Corchado, J.M., Pavón, J., Corchado, E., Castillo, L.F.: Development of CBR-BDI agents: A tourist guide application. In: *Proceedings of the Seventh European Conference on Case-Based Reasoning*, Springer (2004) 547–559

18. Ontañón, S., Mishra, K., Sugandh, N., Ram, A.: Case-based planning and execution for real-time strategy games. In: *Proceedings of the Seventh International Conference on Case-Based Reasoning*, Springer (2007) 164–178
19. Rubin, J., Watson, I.: On combining decisions from multiple expert imitators for performance. In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, AAAI Press (2011) 344–349
20. Floyd, M.W., Esfandiari, B., Lam, K.: A case-based reasoning approach to imitating RoboCup players. In: *Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference*, AAAI Press (2008) 251–256
21. Zarka, R., Cordier, A., Egyed-Zsigmond, E., Lamontagne, L., Mille, A.: Similarity measures to compare episodes in modeled traces. In: *Proceedings of the Twenty-First International Conference on Case-Based Reasoning*, Springer (2013) 358–372
22. Sánchez-Marrè, M., Cortés, U., Martínez, M., Comas, J., Rodríguez-Roda, I.: An approach for temporal case-based reasoning: Episode-based reasoning. In: *Proceedings of the Sixth International Conference on Case-Based Reasoning*, Springer (2005) 465–476
23. Borrajo, D., Roubířková, A., Serina, I.: Progress in case-based planning. *ACM Computing Surveys* **47**(2) (2015) 1–39
24. Jensen, B., Karneeb, J., Borck, H., Aha, D.: Integrating AFSIM as an internal predictor. Technical Report AIC-14-172, Naval Research Laboratory, Navy Center for Applied Research in Artificial Intelligence, Washington, DC (2014)